Configuring FLEX 10K Devices

August 1998, ver. 1.01 Application Note 59

Introduction

FLEX® 10K devices can be configured using one of four configuration schemes, which are ideal for a variety of systems. You can configure FLEX 10K devices with either an EPC1 Configuration EPROM or a microprocessor. See Table 1.

Table 1. FLEX 10K Configuration Schemes

Configuration Scheme	Typical Use
Configuration EPROM	Configuration with the EPC1 Configuration EPROM
Passive serial	Configuration with a serial synchronous microprocessor interface, the BitBlaster™, or the FLEX Download Cable.
Passive parallel synchronous	Configuration with a parallel synchronous microprocessor interface.
Passive parallel asynchronous	Configuration with a parallel asynchronous microprocessor interface. In passive parallel asynchronous configuration, the microprocessor treats the FLEX 10K device as memory.

This application note discusses how to configure for one or more FLEX 10K devices. It should be used together with the *FLEX 10K Embedded Programmable Logic Family Data Sheet* and the *EPC1 Configuration EPROM Data Sheet*. If appropriate, illustrations in this application note show devices with generic "FLEX 10K" labels to indicate they are valid for all FLEX 10K devices.

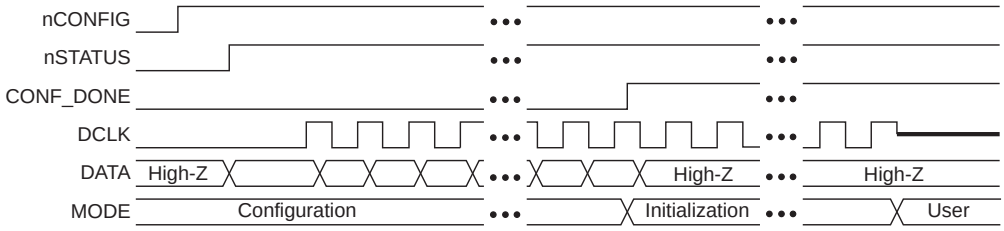
The following topics are discussed:

- Device configuration overview
- FLEX 10K device configuration schemes
- Device options
- Device configuration pins
- Device configuration files
- Device configuration and programming
- Configuration reliability

Device Configuration Overview

During device operation, FLEX 10K devices store configuration data in SRAM cells. Because SRAM memory is volatile, the SRAM cells must be loaded with configuration data each time the device powers up. After the FLEX 10K device is configured, its registers and I/O pins must be initialized. After initialization, the device enters user mode for in-system operation. Figure 1 shows the state of the device during configuration, initialization, and user modes.

Figure 1. FLEX 10K Configuration Cycle



The configuration data for a FLEX 10K device can be loaded using an active or passive configuration scheme. When configuring a FLEX 10K device with an active scheme using an EPC1 Configuration EPROM, control and synchronization signals are generated by the FLEX 10K device and the EPC1. The EPC1 provides the configuration data to the FLEX 10K device. In passive configuration schemes (i.e., configuration with a microprocessor), the FLEX 10K device is incorporated into a system with an intelligent host, such as a microprocessor, that controls the configuration process. The host supplies configuration data from a storage device, e.g., a hard disk, RAM, or other system memory. With passive configuration, the functionality of the FLEX 10K device can be changed while the system is in operation.

The configuration scheme is chosen by driving the FLEX 10K device's `MSEL0` and `MSEL1` pins either high or low, as shown in Table 2. The value of the `MSEL0` and `MSEL1` pins can be changed between configurations to change the configuration mode. However, these pins are most commonly tied to `VCC` or `GND`.

Table 2. Configuration Schemes

MSEL1	MSEL0	Configuration Scheme
0	0	Configuration EPROM or passive serial
1	0	Passive parallel synchronous
1	1	Passive parallel asynchronous

Device option bits and device configuration pins are discussed in this application note on pages 15 and 17, respectively.

Table 3 summarizes the configuration file size required for each FLEX 10K device. To calculate the amount of data storage space required for multi-device configurations, simply add the file sizes for each FLEX 10K device in the design.

Table 3. FLEX 10K Device Configuration File Sizes

Device	Data Size (bits)	Data Size (Kbytes)
EPF10K10	115,000	15
EPF10K20	225,000	28
EPF10K30	368,000	45
EPF10K40	488,000	60
EPF10K50	609,000	75
EPF10K70	881,000	108
EPF10K100	1,172,000	144

The EPC1 Configuration EPROM is a 1-Mbit EPROM; therefore, the EPF10K100 requires two EPC1 Configuration EPROMs for active configuration. The EPC1 can configure multiple smaller FLEX 10K devices, such as eight EPF10K10 devices.

FLEX 10K Device Configuration Schemes

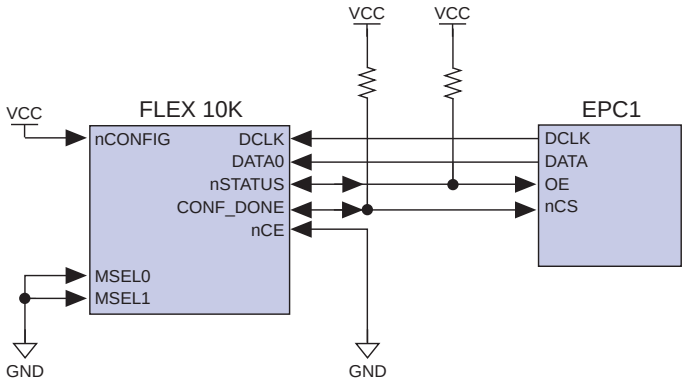
This section describes how to configure FLEX 10K devices with the following configuration schemes:

- Configuration EPROM
- Passive serial
- Passive parallel synchronous
- Passive parallel asynchronous

Configuration EPROM Configuration

The Configuration EPROM scheme uses an Altera-supplied serial EPC1 Configuration EPROM to supply data to the FLEX 10K device in a serial bitstream. See Figure 2.

Figure 2. Configuration EPROM Scheme Circuit

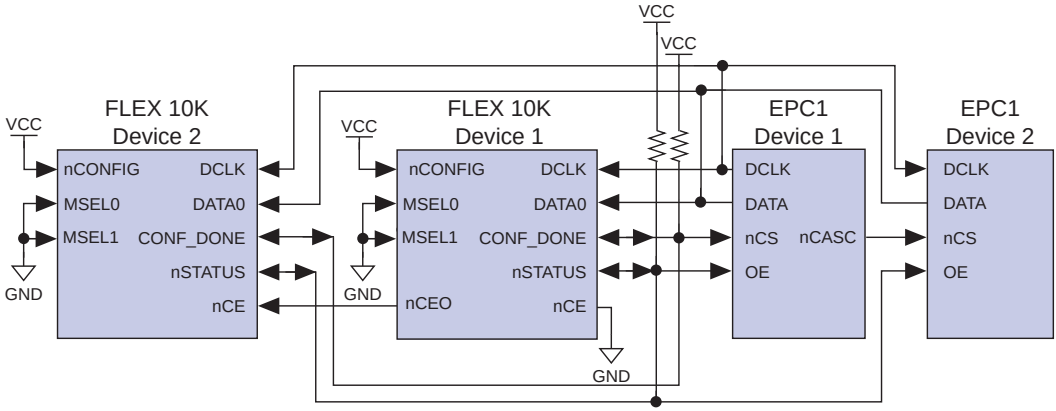


In the Configuration EPROM scheme, nCONFIG is usually tied to VCC. Upon device power-up, the FLEX 10K device senses the low-to-high transition on nCONFIG, which initiates configuration. The FLEX 10K device drives the open-drain CONF_DONE pin low, which drives the nCS pin on the EPC1 low. The open-drain nSTATUS pin is released by the FLEX 10K device and pulled high to enable the EPC1. The EPC1 then uses its internal oscillator to serially clock data from its EPROM cells to the FLEX 10K device.

If an error occurs during configuration, the FLEX 10K device drives the nSTATUS pin low, resetting the EPC1 and internally resetting the FLEX 10K device. If the *Auto-Restart Configuration on Frame Error* option—available from the **Global Project Device Options** dialog box (Assign menu)—is turned on in the MAX+PLUS® II software, the FLEX 10K device automatically reconfigures if an error occurs. If this option is turned off, the outside system must monitor nSTATUS to detect an error and then pulse nCONFIG low to restart configuration. The outside system can pulse nCONFIG if nCONFIG is under system control rather than tied to VCC. When configuration is complete, the FLEX 10K device releases CONF_DONE, which removes the EPC1 from the circuit.

Multiple FLEX 10K devices can be configured with the Configuration EPROM scheme. See Figure 3.

Figure 3. Configuration EPROM Scheme Multi-Device Configuration Circuit



The circuit in Figure 3 is similar to the Configuration EPROM scheme circuit for a single device, except the FLEX 10K devices are cascaded for multi-device configuration. After the first FLEX 10K device has been configured, the nCEO pin on the first device activates the nCE pin on the second device, prompting the second device to begin configuration. The CONF_DONE pins on all the FLEX 10K devices are tied together. Therefore, the FLEX 10K devices initialize and enter user mode at the same time. Additionally, the nSTATUS lines are tied together; if any device detects an error, the entire chain is reset for automatic reconfiguration.

EPC1 Configuration EPROMs can also be cascaded for active configuration of several smaller FLEX 10K devices (e.g., EPF10K10) or one EPF10K100 device. When all the data from the first EPC1 device has been sent, the EPC1 drives nCASC low, which drives nCS on the next EPC1. Less than one clock cycle is required for one EPC1 device to activate the next EPC1 for configuration. The stream of data is uninterrupted.

The waveform in Figure 4 shows the timing waveform for the Configuration EPROM scheme.

Figure 4. Configuration EPROM Scheme Timing Waveform

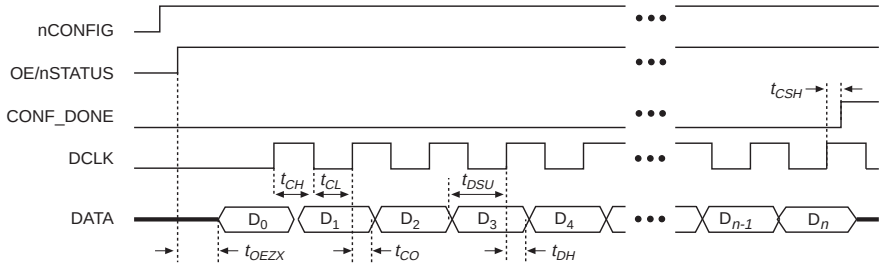


Table 4 defines the timing parameters for the Configuration EPROM scheme.

Table 4. Configuration EPROM Scheme Timing Parameters

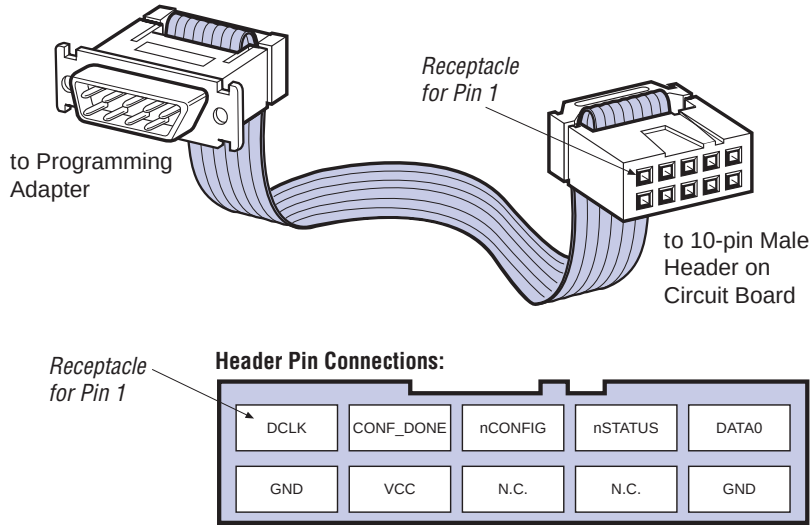
Symbol	Parameter	Min	Max	Units
t_{OEZX}	OE high to DATA output enabled		160	ns
t_{CH}	DCLK high time	50	250	ns
t_{CL}	DCLK low time	50	250	ns
t_{DSU}	Data setup time before rising edge on DCLK	30		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CO}	DCLK to DATA out		30	ns
t_{OEW}	OE low pulse width to guarantee counter reset	100		ns
t_{CSH}	nCS low hold time after DCLK rising edge	0		ns
f_{MAX}	DCLK frequency	2	10	MHz

Passive Serial Configuration

In passive serial (PS) configuration, the BitBlaster, FLEX Download Cable, or microprocessor generates a low-to-high transition on the nCONFIG pin. The microprocessor or programming hardware then places the configuration data on the DATA0 pin of the FLEX 10K device one bit at a time. The data is clocked into the FLEX 10K device until CONF_DONE goes high. The microprocessor or programming hardware must present the least significant bit (LSB) of each byte of data to the FLEX 10K device first. After CONF_DONE goes high, DCLK must be clocked an additional 10 times to initialize the device. The microprocessor must be configured to supply the extra clock cycles to the FLEX 10K device. Device initialization is performed automatically by the BitBlaster serial download cable or the FLEX Download Cable programming hardware (see Figure 5).

Handshaking signals are not used in PS configuration modes. Therefore, the configuration clock speed must be below 10 MHz.

Figure 5. FLEX Download Cable Signals & Positions



For more information on how to use the BitBlaster, go to the *BitBlaster Serial Download Cable Data Sheet* in the ***1995 Data Book***.

Figure 6 shows the configuration circuit for PS configuration with a microprocessor.

Figure 6. PS Configuration Circuit with Microprocessor

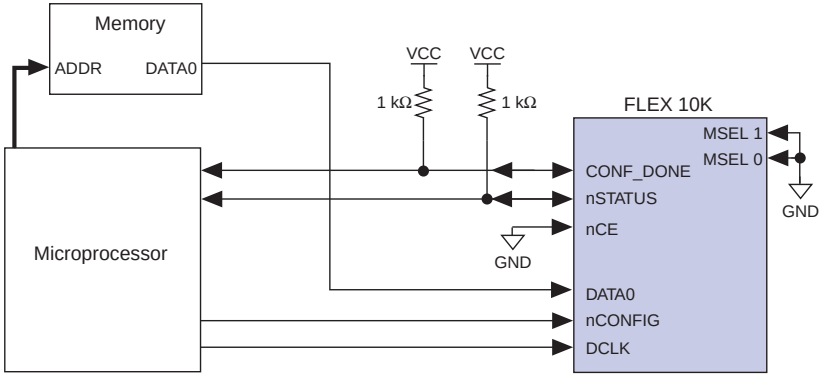
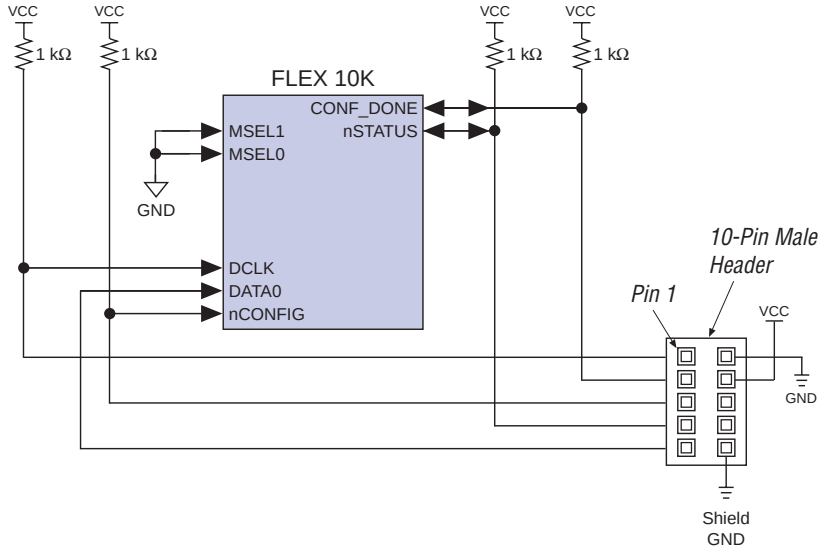


Figure 7 shows the configuration circuit for PS configuration with programming hardware.

Figure 7. PS Configuration Circuit with Programming Hardware



For multi-device PS configuration, the nCEO pin on the first FLEX 10K device is cascaded to the nCE pin of the next device. The second FLEX 10K device begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the microprocessor. See Figure 8.

Figure 8. PS Multi-Device Configuration Circuit

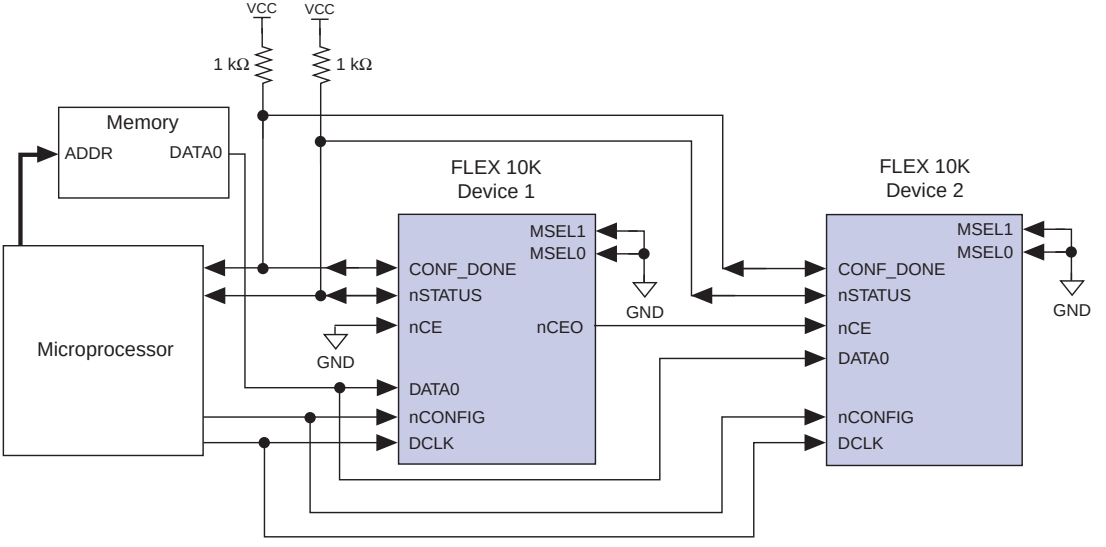


Figure 9 shows the timing waveform for PS configuration.

Figure 9. PS Timing Waveform

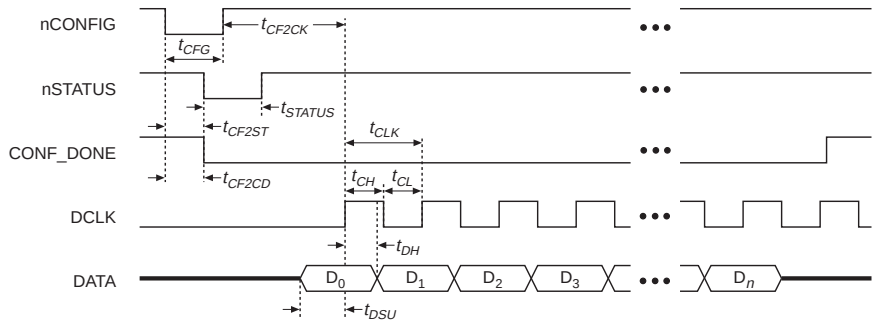


Table 5 defines the timing parameters for PS configuration. .

Table 5. PS Timing Parameters

Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		1	µs
t_{CF2ST}	nCONFIG low to nSTATUS low		1	µs
t_{CFG}	nCONFIG low pulse width	2		µs
t_{STATUS}	nSTATUS low pulse width	2.5		µs
t_{CF2CK}	nCONFIG high to first rising edge on DCLK *(1)*	5		µs
t_{DSU}	Data setup time before rising edge on DCLK	30		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	50		ns
t_{CL}	DCLK low time	50		ns
t_{CLK}	DCLK period	100		ns
f_{MAX}	DCLK maximum frequency		10	MHz

Note:

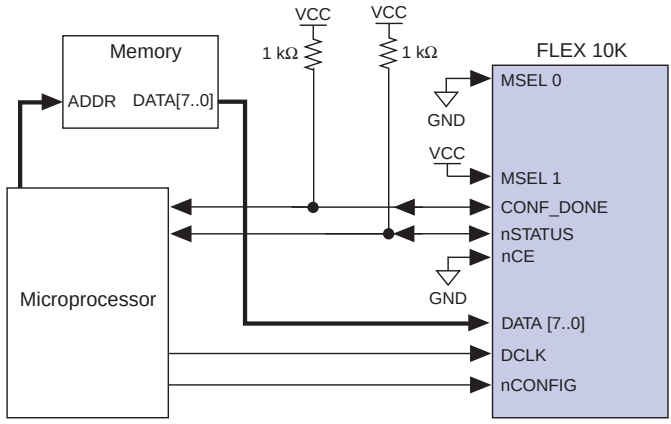
(1) DCLK can be applied before nCONFIG goes high. However, it will be ignored until 5 µs after nCONFIG goes high.

Passive Parallel Synchronous Configuration

In passive parallel synchronous (PPS) configuration, nCONFIG is controlled by an intelligent host, such as a microprocessor. To begin configuration, nCONFIG is given a low-to-high transition and the microprocessor places an 8-bit configuration word on the data inputs of the FLEX 10K device. The microprocessor clocks the FLEX 10K device. On the first rising clock edge, a byte of configuration data is latched into the FLEX 10K device; the subsequent 8 falling clock edges serialize the data in the device.

On the ninth rising clock edge, the next byte of configuration data is latched and serialized into the FLEX 10K device. If an error occurs during configuration, the FLEX 10K nSTATUS pin drives low. The microprocessor senses this low signal and begins reconfiguration or issues an error. Once the FLEX 10K device has been successfully configured, the CONF_DONE pin is released by the FLEX 10K device, indicating that configuration is complete. The DCLK pin must be clocked 10 times after CONF_DONE is released to initialize the device. See Figure 10.

Figure 10. PPS Configuration Circuit



PPS configuration can also be used to configure multiple FLEX 10K devices. In multi-device PPS configuration, the FLEX 10K devices are cascaded. Once the first FLEX 10K device is configured, it drives nCEO low, which drives nCE on the second device low. The second FLEX 10K device begins configuration within one clock cycle. The FLEX 10K CONF_DONE pins are tied together so the devices initialize and enter user mode at the same time. In addition, the nSTATUS signals are tied together; if any device detects an error, the entire chain is reset for automatic reconfiguration. See Figure 11.

Figure 11. PPS Multi-Device Configuration Circuit

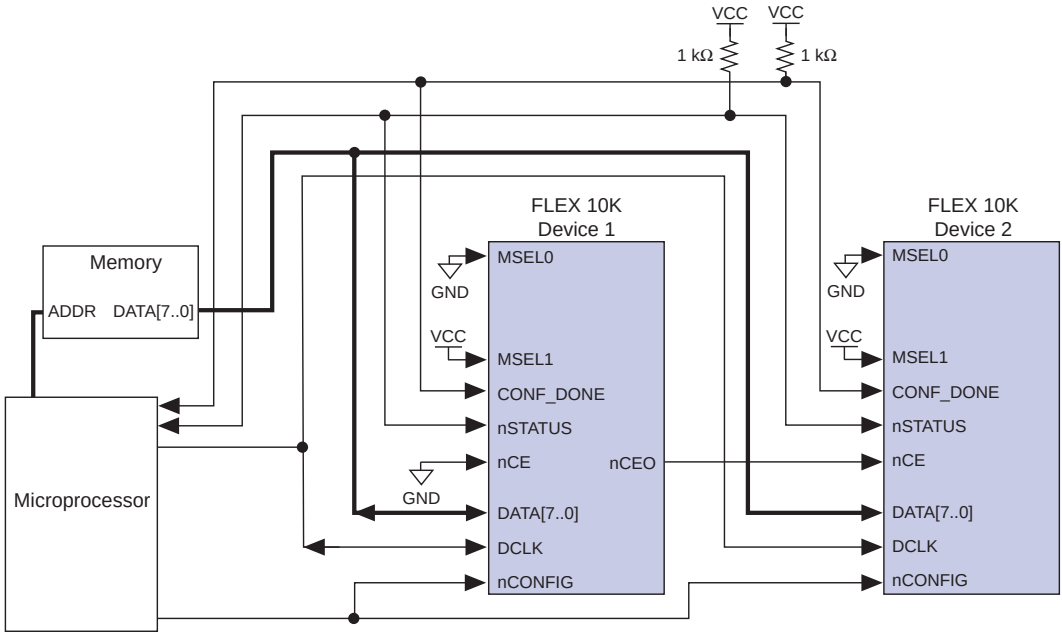


Figure 12 shows the timing waveform for PPS configuration.

Figure 12. PPS Timing Waveform

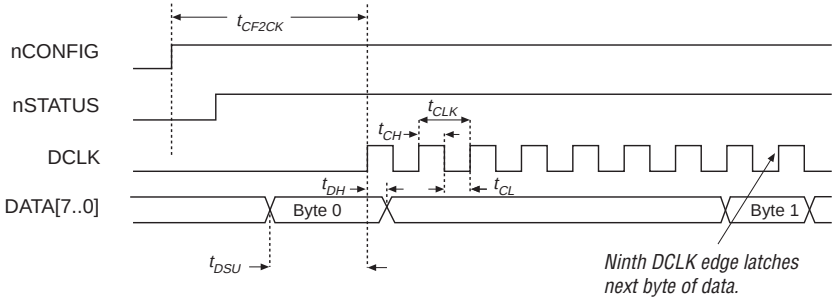


Table 6 defines the timing parameters for PPS configuration.

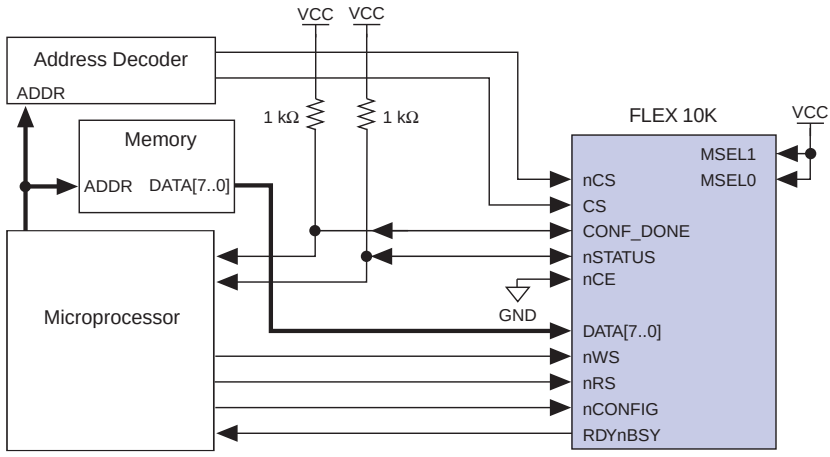
Table 6. PPS Timing Parameters

Symbol	Parameter	Min	Max	Units
t_{CF2CK}	nCONFIG high to first rising edge on DCLK	5		µs
t_{DSU}	Data setup time before rising edge on DCLK	30		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	50		ns
t_{CL}	DCLK low time	50		ns
t_{CLK}	DCLK period	100		ns
f_{MAX}	DCLK frequency		6	MHz

Passive Parallel Asynchronous Configuration

In passive parallel asynchronous (PPA) configuration, nCONFIG is controlled by a microprocessor. To begin configuration, the microprocessor drives nCONFIG high. The microprocessor then asserts the nCS and CS inputs to the FLEX 10K device. The microprocessor places an 8-bit configuration word on the data inputs of the FLEX 10K device and pulses nWS low on the FLEX 10K device. On the rising edge of the low pulse on nWS, the FLEX 10K device latches the byte of configuration data. The FLEX 10K device then drives the RDYnBSY signal low, indicating that it is processing the byte of configuration data. The microprocessor can then perform other system functions while the FLEX 10K devices is processing the byte of data. Configuration can be paused by de-asserting the nCS or CS pins on the FLEX 10K device. Figure 13 shows the PPA configuration circuit. An address decoder controls nCS and CS on the FLEX 10K device. This decoder allows the microprocessor to select the FLEX 10K device by accessing a particular address, simplifying the configuration process.

Figure 13. PPA Configuration Circuit



The FLEX 10K device can internally serialize data without the microprocessor. When the FLEX 10K device is ready for the next byte of configuration data, it drives RDYnBSY high. The microprocessor polls the RDYnBSY signal and when it senses a high signal it strobes the next byte of configuration data into the FLEX 10K device. Alternatively, the nRS signal can be strobed, causing the RDYnBSY signal to appear on DATA7. Reading the state of the configuration data by strobing nRS permits the system to save an I/O port if necessary. Data should not be driven onto the data bus while nRS is low because this will cause contention on DATA7. If the nRS pin is not used to monitor configuration, it should be tied high.

PPA mode can also be used to configure multiple FLEX 10K devices. Multi-device PPA configuration is similar to single-device PPA configuration, except the FLEX 10K devices are cascaded. After the first FLEX 10K device is configured, it drives nCEO low, which drives the nCE pin on the second FLEX 10K device low, causing it to begin configuration. The second FLEX 10K device begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the microprocessor. All FLEX 10K device CONF_DONE pins are tied together, so all FLEX 10K devices initialize and enter user mode at the same time. See Figure 14.

Figure 14. PPA Multi-Device Configuration Circuit

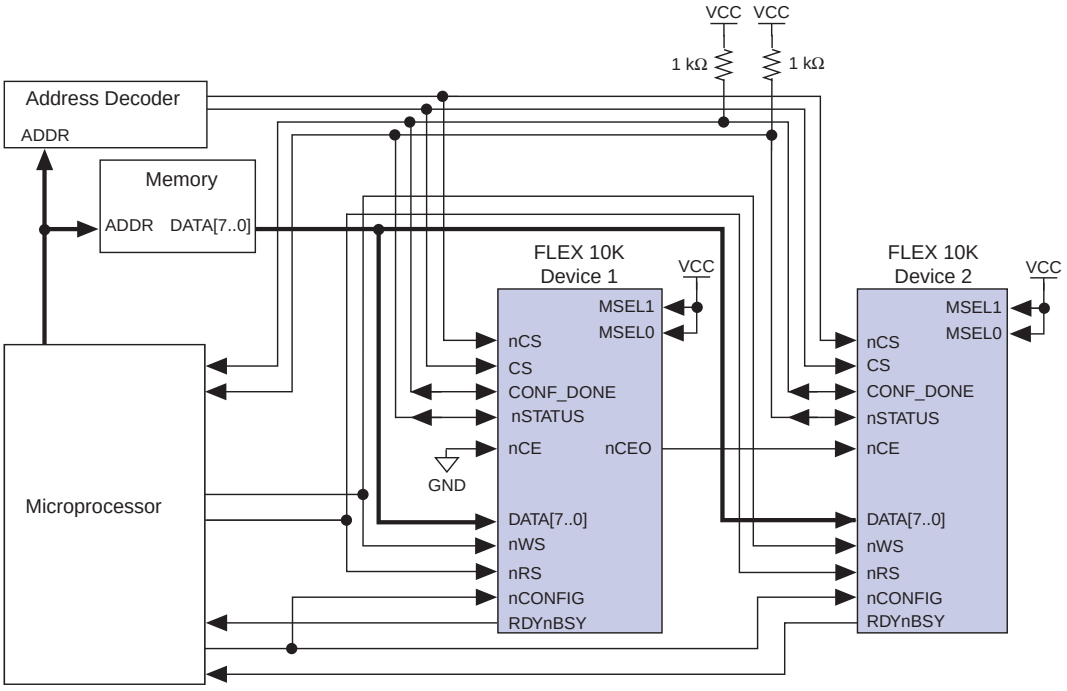


Figure 15 shows the timing waveform for PPA configuration.

Figure 15. PPA Timing Waveform

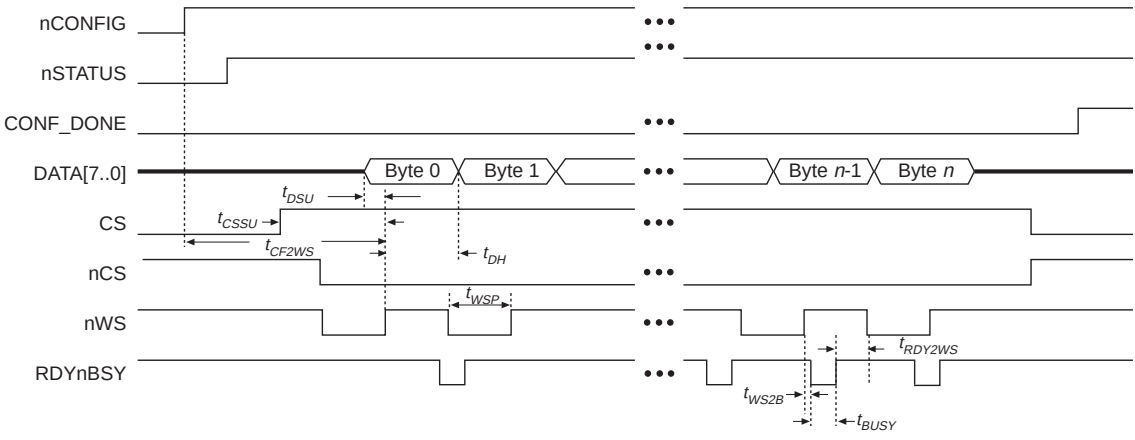


Figure 16 shows the timing waveform for a strobed nRS signal.

Figure 16. PPA Timing Waveform Using nRS and nWS

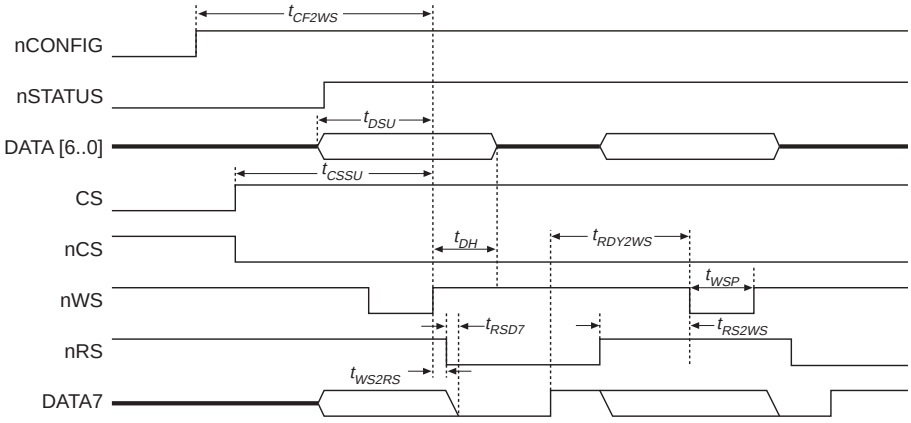


Table 7 summarizes the timing parameters for PPA configuration.

Table 7. PPA Timing Parameters

Symbol	Parameter	Min	Max	Units
t_{CF2WS}	nCONFIG high to first rising edge on nWS	5		µs
t_{DSU}	Data setup time before rising edge on nWS	50		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip Select setup time before rising edge on nWS	50		ns
t_{WSP}	nWS low pulse width	200		ns
t_{WS2B}	nWS rising edge to RDYnBSY low		50	ns
t_{BUSY}	RDYnBSY low pulse width		2	µs
t_{RDY2WS}	RDYnBSY rising edge to nWS falling edge	50		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	200		ns
t_{RS2WS}	nRS rising edge to nWS falling edge	200		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		50	ns

Device Options

You can set FLEX 10K device operation options in the Altera® MAX+PLUS II development software by choosing **Global Project Device Options** (Assign menu). Table 8 summarizes each of these options.

Table 8. FLEX 10K Configuration Option Bits (Part 1 of 2)

Device Option	Option Usage	Default Configuration (Option Off)	Modified Configuration (Option On)
User-Supplied Start-Up Clock	The FLEX 10K devices must be clocked 10 times after it is configured to initialize the device. The user can choose the clock source.	In the PPA configuration scheme, the internal FLEX 10K oscillator supplies the initialization clock. In Configuration EPROM, PS, and PPS configuration, the internal oscillator is disabled. Therefore, external circuitry must provide the initialization clock on the DCLK pin. In the Configuration EPROM scheme, the EPC1 supplies the clock; in PS and PPS configuration, the microprocessor supplies the clock.	The user provides the clock on the CLKUSR pin. This clock can be used to synchronize the initialization of multiple FLEX 10K devices.
Auto-Restart Configuration on Frame Error	If a data error occurs during FLEX 10K device configuration, the user can choose how to restart the configuration.	The configuration process stops until the user directs the device to restart configuration. nSTATUS is driven low when there is an error. When nCONFIG is pulled low and then high, the device begins to reconfigure.	The configuration process will automatically restart. The nSTATUS pin drives low for 10 clock cycles and then releases. The nSTATUS pin is then pulled to V_{CC}, indicating that the configuration process has started. In the Configuration EPROM scheme, the nSTATUS reset pulse automatically resets the EPC1 Configuration EPROM if the nSTATUS pin on the FLEX 10K device is tied to OE on the Configuration EPROM. If an error occurs during passive configuration, the device can be reconfigured without the system having to pulse nCONFIG.

Table 8. FLEX 10K Configuration Option Bits (Part 2 of 2)

Device Option	Option Usage	Default Configuration (Option Off)	Modified Configuration (Option On)
Release Clears before Tri-States	During configuration, the I/O pins on the device are tri-stated. The user can choose the order in which to release the tri-states and clear the registers during initialization.	Directs the device to release the tri-states on the I/O pins of the device before releasing the Clear signal on the device's registers.	Directs the device to release the Clear signals on its registers before releasing the tri-states.
Enable Chip-Wide Reset	Enables a single pin to reset every register on the FLEX 10K device.	Chip-Wide Reset is not enabled. The DEV_CLRn pin is available as a user I/O pin.	Chip-Wide Reset is enabled for all registers within the device. All registers are cleared when the DEV_CLRn pin is driven low.
Enable Chip-Wide Output Enable	Enables a single pin to control all of the tri-states on a FLEX 10K device.	Chip-Wide Output Enable is not enabled. The DEV_OE pin is available as a user I/O pin.	Chip-Wide Output Enable is enabled for all tri-states on the FLEX 10K device. After configuration, all user I/O pins are tri-stated when DEV_OE is low.
Enable INIT_DONE Output	Enables a pin to drive out a signal when the initialization process is complete and the device has entered user mode.	The INIT_DONE signal is not available. The INIT_DONE pin is available as a user I/O pin.	The INIT_DONE signal is available on the open-drain INIT_DONE pin. This pin drives low during configuration. After initialization, it is released and pulled high externally.

Device Configuration Pins

Each FLEX 10K device has 9 dedicated configuration pins and 14 dual-purpose configuration pins, some or all of which are used in the configuration schemes discussed in this application note. Table 9 summarizes the FLEX 10K configuration pins.

Table 9. Pin Functions (Part 1 of 2)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
MSEL0 MSEL1	–	All	Input	2-bit configuration input. Informs FLEX 10K device of the configuration scheme. These bits are set as shown in Table 3 earlier in this application note.
nSTATUS	–	All	Bidirectional open-drain	The FLEX 10K device drives nSTATUS low immediately after power-up and releases it within 100 ms. The nSTATUS pin must be pulled up to V_{CC} with a 1-kΩ resistor. If an error occurs during configuration, nSTATUS is pulled low by the FLEX 10K device. If an external source drives the nSTATUS pin low (as in multi-device configuration), the FLEX 10K device enters an error state.
nCONFIG	–	All	Input	Configuration control input. A low resets the FLEX 10K device. A low-to-high transition begins configuration.
CONF_DONE	–	All	Bidirectional open drain	Status output. The CONF_DONE pin is driven low by the FLEX 10K device during configuration. After all configuration data has been received without errors, the FLEX 10K device tri-states CONF_DONE. Status input. A high on this input directs the FLEX 10K device to initialize and enter user mode. The CONF_DONE net must be pulled to V_{CC} with a 1.0-kΩ resistor and may be driven low by an external source to delay the initialization process.
DCLK	–	Configuration EPROM, PPS, PS	Input	Clock input used to clock data from an external source into the FLEX 10K device.
nCE	–	All	Input	Active-low Chip Enable. The nCE pin activates the device with a low signal to allow configuration and should be tied low for single device configuration.
nCEO	–	Multi-device	Output	Output that drives low when FLEX 10K configuration is complete. This pin feeds the nCE of another FLEX 10K device in a multi-device, cascaded configuration scheme.
nWS	I/O	PPA	Input	Write strobe input. A low-to-high transition causes the FLEX 10K device to latch a byte of data on the DATA[7..0] pins.
nRS	I/O	PPA	Input	Read strobe input. A low input directs the FLEX 10K device to drive the RDYnBSY signal on the DATA7 pin. If the nRS pin is not used, it should be tied high.

Table 9. Pin Functions (Part 2 of 2)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
RDYnBSY	I/O	PPA	Output	Ready output. A high output indicates that the FLEX 10K device is ready to accept another byte of data. A low output indicates that the FLEX 10K device is not ready to receive another byte of data.
nCS CS	I/O	PPA	Inputs	Chip-Select inputs. A low on nCS and a high on CS selects the FLEX 10K device for configuration. If only one Chip-Select input is used, the other must be tied to the active value (e.g., nCS can be tied to ground if CS is used).
CLKUSR	I/O	All	Input	Optional user-supplied clock input. Synchronizes initialization of one or more FLEX 10K devices.
DATA[7..0]	I/O	PPS, PPA	Inputs	Data inputs. Byte-wide configuration data is presented to the FLEX 10K device on all eight data pins.
DATA0	–	Configuration EPROM, PS	Input	Data input. Bit-wide configuration data is presented to the FLEX 10K device on the DATA0 pin.
DATA7	I/O	PPA	Output	In the PPA configuration scheme, the DATA7 pin presents the RDYnBSY signal after the nRS signal has been strobed, which may be more convenient for microprocessors than using the RDYnBSY pin.
INIT_DONE	I/O	All	Output open-drain	Status pin. Can be used to indicate when the device has initialized and is in user mode. The INIT_DONE pin will drive low during configuration. Before and after configuration the INIT_DONE pin is released and is pulled to V_{CC} by an external pull-up resistor. Because INIT_DONE is tri-stated before configuration, it will be pulled high by the external pull-up resistor. Therefore, it is important that the monitoring circuitry be able to detect a low-to-high transition. This option is set in MAX+PLUS II.
DEV_OE	I/O	All	Input	Optional pin that allows the user to override all tri-states on the device. When this pin is driven low all I/Os are tri-stated; when this pin is driven high, all I/Os behave as in the user design. This option is set in MAX+PLUS II.
DEV_CLRn	I/O	All	Input	Optional pin that allows the user to override all clears on all registers on the device. When this pin is driven low all registers are cleared; when this pin is driven high, all registers behave as in the user design. This option is set in MAX+PLUS II.

Device Configuration Files

Altera's MAX+PLUS II development tools can create one or more configuration and programming files to support all configuration schemes discussed in this application note. This section describes these files.

SRAM Object File (.sof)

An SOF is used in PS configuration when the data is downloaded directly from the Altera programming hardware with the FLEX Download Cable. The MAX+PLUS II Compiler's Assembler module automatically creates the SOF for each FLEX 10K device in your design. MAX+PLUS II controls the configuration sequence and automatically inserts the appropriate headers into the configuration data stream. All other configuration files are created from the SOF.

Programming Object File (.pof)

A POF is used by the Altera programming hardware to program an EPC1 Configuration EPROM, which is used to configure FLEX 10K devices in Configuration EPROM mode. A POF is automatically generated when each FLEX 10K project is compiled. Each EPC1 requires a POF for programming. For smaller FLEX 10K devices (e.g., EPF10K20) multiple POFs can fit into one EPC1; for the EPF10K100 or several large FLEX 10K devices (e.g., two EPF10K70 devices), two or more EPC1 devices are required to hold the configuration data.

Serial Bitstream File (.sbf)

An SBF is used to configure FLEX 10K devices in-system in PS mode with the BitBlaster serial download cable. For information on how to use the BitBlaster, refer to the *BitBlaster Serial Download Cable Data Sheet*.

To create an SBF, choose the **Combine Programming Files** command (File menu) in the MAX+PLUS II Compiler or Programmer. Add the SOF files you wish to combine to the *Selected Files* box by selecting each file in the *Directories* box and choosing **Add**. Then, choose *.sbf (Sequential)* in the *File Format* box and choose **OK**. For more information on creating SBFs, search for "SBF" in MAX+PLUS II Help.

Hexadecimal (Intel-Format) File (.hex)

A Hex File is an ASCII file in the Intel Hex format. This file is used by third-party programmers, such as Data I/O programmers, to program Altera's serial EPC1 Configuration EPROM. Hex Files are also used to

program parallel EPROMs with third-party programming hardware. You can use parallel EPROMs in the PPS and PPA configuration schemes, in which a microprocessor uses the parallel EPROM as the data source.

To create a Hex File, choose the **Combine Programming Files** command (File menu) in the MAX+PLUS II Compiler or Programmer. Add one or more SOF files to the *Selected Files* box by selecting the files in the *Directories* box and choosing **Add**. Then, in the *File Format* box, choose *.hex (single-device)*. Choose **OK**. For more information on creating Hex Files, search for "Hex File" in MAX+PLUS II Help.

Tabular Text File (.ttf)

The TTF is a tabular ASCII file that provides a comma-separated version of the configuration data for the PPA, PPS, and bit-wide PS configuration schemes. In some applications, the storage device that contains the FLEX 10K configuration data is neither dedicated to nor connected directly to the FLEX 10K device. For example, an EPROM can also contain executable code for a system (e.g., BIOS routines) and other data. The TTF allows you to include the FLEX 10K configuration data as part of the source code for the microprocessor using "include" or "source" commands. The microprocessor can access this data from an EPROM or a mass-storage device and load it into the FLEX 10K device.

A TTF can be imported into nearly any assembly language or high-level language compiler. Consult the documentation for your compiler or assembler for information on including other source files.

To create a TTF, choose the **Combine Programming Files** command (File menu) in the MAX+PLUS II Compiler or Programmer. Add one or more SOF files to the *Selected Files* box by selecting the files in the *Directories* box and choosing **Add**. Then, in the *File Format* box, choose *.ttf (Sequential)*. Choose **OK**. For more information on creating TTFs, search for "TTF" in MAX+PLUS II Help.

Raw Binary File (.rbf)

The RBF is a binary file containing the FLEX 10K configuration data (e.g., 85 becomes 10000101). Data must be stored so that the least significant bit (LSB) of each byte of data is loaded first. The converted image can be stored on a mass storage device. The microprocessor can then read data from the binary file and load it into the FLEX 10K device. You can also use the microprocessor to perform real-time conversion during configuration. In the PPA and PPS configuration schemes, the FLEX 10K device receives

its information in parallel from the data bus, a data port on the microprocessor, or some other byte-wide channel. In the bit-wide PS configuration scheme, the data is shifted in serially.

To create an RBF, choose the **Combine Programming Files** command (File menu) in the MAX+PLUS II Compiler or Programmer. Add one or more SOF files to the *Selected Files* box by selecting the files in the *Directories* box and choosing **Add**. Then, in the *File Format* box, choose *.rbf (Sequential)*. Choose **OK**. For more information on creating RBFs, search for "RBF" in MAX+PLUS II Help.

Device Programming & Configuration

You can configure a FLEX 10K device with data from an EPC1 Configuration EPROM, or you can download data into the FLEX 10K device with the MAX+PLUS II software.

Programming a Configuration EPROM

You can program the EPC1 Configuration EPROM with MAX+PLUS II, the PL-MPU Master Programming Unit, and the appropriate Configuration EPROM programming adapter. The PLMJ1213 adapter programs EPC1 Configuration EPROMs in 8-pin plastic dual in-line packages (PDIP) and 20-pin plastic J-lead chip carrier (PLCC) packages.

To program an Altera EPC1 Configuration EPROM:

1. Choose the **Programmer** command (MAX+PLUS II menu) to open the Programmer window.
2. By default, the Programmer loads the POF for the current project. If necessary, load a different POF with the **Select Programming File** command (File menu). The appropriate device for the current programming file is displayed in the Device field.
3. Insert a blank Configuration EPROM into the 8-pin DIP or 20-pin J-lead socket on the programming adapter.
4. Choose the **Program** button.

After successful programming, you can place the EPC1 Configuration EPROM on the target board to configure a FLEX 10K device in the Configuration EPROM scheme.

For more information on the EPC1, refer to the *EPC1 Configuration EPROM for FLEX Devices Data Sheet*.

Configuration with MAX+PLUS II & the FLEX Download Cable

You can configure a FLEX 10K device in-circuit with MAX+PLUS II and the FLEX Download Cable:

1. Connect the FLEX Download Cable to the 9-pin D-type connector on a Configuration EPROM programming adapter.
2. Connect the other end of the FLEX Download Cable to the 10-pin male header on the target board.
3. Start MAX+PLUS II and choose the **Programmer** command (MAX+PLUS II menu) to open the Programmer window.
4. Choose the **Select Programming File** command (File menu).
5. Select the desired SOF name in the *Files* box or type a name in the *File Name* box. If you choose a programming file from another project, you are asked if you wish to change the current project name.
6. Choose **OK**.
7. Choose the **Program** button to configure the device.

After the device is configured and initialized, it enters user mode and operates as a logic device. The FLEX Download Cable is electrically removed from the circuit and does not influence circuit operation. You can also physically disconnect the FLEX Download Cable without disturbing the FLEX 10K configuration data or device operation.

Configuration with MAX+PLUS II & the BitBlaster

For instructions on how to configure FLEX devices with the BitBlaster serial download cable, refer to the *BitBlaster Serial Download Cable Data Sheet* in the ***1995 Data Book***.

Configuration Reliability

The FLEX 10K architecture has been designed to minimize the effects of power supply and data noise in a system, and to ensure that the configuration data is not corrupted during configuration or normal user-mode operation. A number of circuit design features are provided to ensure the highest possible level of reliability from this SRAM technology.

Cyclic redundancy code (CRC) circuitry is used to validate every data frame (i.e., sequence of data bits) as it is loaded into the FLEX 10K device. If the CRC generated by the FLEX 10K device does not match the data stored in the data stream, the configuration process is halted, and the nSTATUS pin is pulled and held low to indicate an error condition. CRC circuitry ensures that noisy systems will not cause errors that yield an incorrect or incomplete configuration.

The FLEX 10K architecture also provides a very high level of reliability in low-voltage brown-out conditions. The FLEX 10K devices SRAM cells require a certain V_{CC} level to maintain accurate data. This voltage threshold is significantly lower than the voltage required to activate the power-on reset (POR) circuitry in the FLEX 10K device. Therefore, the FLEX 10K device stops operating if the V_{CC} starts to fail, and indicates an operation error by pulling and holding the nSTATUS pin low. The device must then be reconfigured before it can resume operation as a logic device. In active configuration schemes, reconfiguration begins as soon as V_{CC} returns to an acceptable level provided the nCONFIG pin is tied to V_{CC}. The low pulse on nSTATUS resets the EPROM by driving OE low. In passive configuration schemes, the host system starts the reconfiguration process.

These device features ensure that FLEX 10K devices have the highest possible reliability in a wide variety of environments, and provide the same high level of system reliability that exists in other Altera programmable logic devices.

Revision History

The information contained in *Application Note 59 (Configuring FLEX 10K Devices)* version 1.01 supersedes information published in previous versions. *Application Note 59 (Configuring FLEX 10K Devices)* version 1.01 contains the following changes:

- Added the DCLK signal to Figure 6 on page 7 and Figure 8 on page 8.
- Added note to Table 5 on page 9.
- Updated Figure 15 on page 14.

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com
Applications Hotline:
(800) 800-EPLD
Customer Marketing:
(408) 544-7104
Literature Services:
(888) 3-ALTERA
lit_req@altera.com